

REMARKS

This responds to the Office Action mailed on November 21, 2006, and the references cited therewith.

Claims 1-6 were amended, claims 8-14 are canceled, and claims 15-28 are newly added; as a result, claims 1-7 and 15-28 are now pending in this application. A check for \$25 is enclosed to cover the cost of the additional claims.

§102 Rejection of the Claims

Claims 1-14 were rejected under 35 U.S.C. § 102(b) for anticipation by Lehman, et al. (U.S. 4,796,179, hereinafter, “Lehman”).

Anticipation requires the disclosure in a single prior art reference of each element of the claim under consideration. *In re Dillon* 919 F.2d 688, 16 USPQ 2d 1897, 1908 (Fed. Cir. 1990) (en banc), cert. denied, 500 U.S. 904 (1991). It is not enough, however, that the prior art reference discloses all the claimed elements in isolation. Rather, “[a]nticipation requires the presence in a single prior reference disclosure of each and every element of the claimed invention, *arranged as in the claim.*” *Lindemann Maschinenfabrik GmbH v. American Hoist & Derrick Co.*, 730 F.2d 1452, 221 USPQ 481, 485 (Fed. Cir. 1984) (citing *Connell v. Sears, Roebuck & Co.*, 722 F.2d 1542, 220 USPQ 193 (Fed. Cir. 1983)) (emphasis added).

Applicants respectfully submit that the Office Action did not make out a *prima facie* case of anticipation for at least the following reasons:

The reference does not teach each and every claim element.

To anticipate a claim, the reference must teach every element of the claim. “A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference.” *Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987).

Amended Claim 1 recites:

“A method for developing a real-time operating system, comprising:
a) specifying a set of n tasks, task (1) through task(n), to be scheduled for execution,
at least one of the tasks of said set of n tasks being selectively configurable as a preemptive or a non-preemptive task;

- b) specifying an a scheduling algorithm for scheduling the execution of said set of n tasks; and
- c) synthesizing source code with embedded commands to implement a task scheduler that uses said scheduling algorithm and said embedded commands for controlling execution of said set of n tasks."

(Emphasis Added)

Lehman discloses:

FIG. 1 depicts the relationship between the automatic code generator 20 of the present invention and a real time multirate system 22. The goal of the code generator 20 is to generate computer software 24 which will enable a multirate, digital, real time control system 26 to control a system 22 in accordance with a specification (called a functional description) 28 generated by the user 30 of the code generator 20.

The functional description 28 is a database for representing the computational relationships between the control system's input and output signals. These computational relationships are defined in terms of functional blocks from a library 31 of such functional blocks. Each functional block in the library 31 defines one or more logical and/or algebraic computations, and the specified combination of functional blocks defines the set of computations which are to be performed by the control system.

In the preferred embodiment, the automatic code generator 20 is a computer with a terminal and software for converting the functional description 28 into code 24 for the control system 26. The major elements of the code generator's software are a linker 32 which generates a second database, called a linked database 34, and a code generator routine 36 which uses both the linked database 34 and the functional description database 28 to generate the code 24 for the control system. The linked database 34 contains information regarding all the connections between the different functional blocks in the control system software, and how these are organized into "subsystems", as described below.

The user builds up a functional description using an authoring program that generates both a functional description database 28 and a corresponding block diagram that the user can view on his computer terminal.

(Lehman at col. 4, lines 63-68 to col. 5, lines 1-31) (Emphasis Added)

Source Code Templated.

Source code templates are sections of source code, or instructions for generating source code, which implement specified computations. In the present invention, each functional block has a corresponding source code template which is used to generate the source code. Each template includes invariant code which defines a computation, and variables for tailoring the source code to use any specified parameters and also to couple the source code to the memory locations for its inputs and outputs.

Preemptive Priority-Based Periodic Scheduling.

In the present invention, the code segments associated with each subsystem (i.e., each specified computational rate) must be executed on a periodic basis or in response to a trigger condition. A software routine generated by the present invention, called the Scheduler, controls not only when the execution of each code segment is initiated, but also which code segments should be executed first when two or more code segments are running concurrently.

Periodic scheduling is conducted according to the time line that specifies when each code segment is to sample its inputs and begin execution. By assigning the subsystems with the fastest repetition rates the highest priority, the present invention allows the code segments with the highest sample rate to preempt the execution of slower subsystems. Lower priority code segments resume execution when no higher priority computations are pending.

Application Specific Operating System.

The software that controls the preemptive priority-based periodic scheduling of the present invention is sometimes referred to herein as an application specific operating system because it performs the functions characteristic of general purpose multitasking operating systems. The present invention can generate a scheduler routine which will actually replace the control system's operating system if that operating system does not include a preemptive priority-based multitasking capability.

Shortest Remaining Time First Scheduling Discipline.

Tasks which need to be completed the soonest (i.e., executed the most rapidly) are assigned the highest execution priority. High priority tasks preempt lower priority tasks running on the same processor. This scheduling methodology, sometimes called a "shortest-remaining-time-first scheduling discipline", maximized utilization of the control system's computing power and helps to insure that all of the control system's tasks are completed on time.

(Lehman at col. 9, lines 42-68 to col. 10, lines 1-22) (Emphasis Added)

In the light of the above quotes, Lehman teaches that “[h]igh priority tasks preempt lower priority tasks running on the same processor”. However, Lehman is silent with respect to, “at least one of the tasks of said set of n tasks being selectively configurable as a preemptive or a non-preemptive task”, as recited in claim 1. Lehman does not teach or suggest selectively configurable tasks. Further, Lehman is silent on, “synthesizing source code with embedded commands to implement a task scheduler that uses said scheduling algorithm and said embedded commands for controlling execution of said set of n tasks”. Lehman does not use embedded commands. Rather, Lehman provides functional blocks where each functional block has a corresponding source code template which is used to generate the source code. Each template in Lehman includes invariant code which defines a computation, and variables for tailoring the source code to use any specified parameters and also to couple the source code to the memory locations for its inputs and outputs. The present invention as currently claimed does not use or need templates or invariant code. As such, Lehman fails to teach each and every element of claim 1; thus, Lehman does not anticipate claim 1. Therefore, Applicants respectfully submit that at least for this reason, independent claim 1 and its dependent claims 2-7 are allowable.

In addition, because new claims 15-28 present similar limitations as in claim 1, Applicants submit that at least for the same reasons set forth above, independent claims 15 and 22 and their respective dependent claims 16-21 and 23-28 are allowable and thus

AMENDMENT AND RESPONSE UNDER 37 CFR § 1.111

Serial Number: 10/688,573

Filing Date: October 20, 2003

Title: SOFTWARE TOOL FOR SYNTHESIZING A REAL-TIME OPERATING SYSTEM

Page 8
Dkt: Zeid-01

their rejections should be withdrawn.

CONCLUSION

Applicant respectfully submits that the claims are in condition for allowance, and notification to that effect is earnestly requested. The Examiner is invited to telephone Applicant's attorney, Jim H. Salter at 408-406-4855 to facilitate prosecution of this application.

Respectfully submitted,

ROBERT M. ZEIDMAN

By his Representatives,

Salter Intellectual Property Law
105 Thoreau Lane
Folsom, CA 95630
408-406-4855

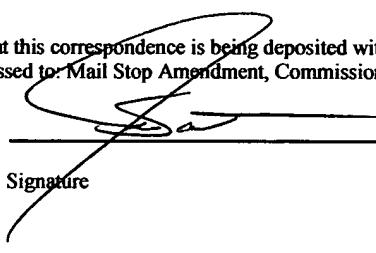
By _____


Jim H. Salter
Reg. No. 35,668

Date January 26, 2007

CERTIFICATE UNDER 37 CFR 1.8: The undersigned hereby certifies that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail, in an envelope addressed to: Mail Stop Amendment, Commissioner of Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on this 26th day of January 2007.

Jim H. SALTER
Name


Signature